REMARKS

The present amendment is prepared in accordance with the new revised requirements of 37 C.F.R. § 1.121. A complete listing of all the claims in the application is shown above showing the status of each claim and the amendments to the specification are shown above in the replacement paragraphs. For current amendments, inserted material is underlined and deleted material has a line therethrough.

Applicants appreciate the thoroughness with which the Examiner has examined the above-identified application. Reconsideration is requested in view of the amendments above and the remarks below.

Claims 1-7 and 9-15 have been amended to clarify that which applicants regard as the invention, as well as to correct for antecedent basis errors. Support for the amendments can be found in the specification at page 8, line 22 to page 19, line 20, and in particular, at page 9, line 10 to page 13, line 10, page 14, lines 9-21, page 15, lines 13-16, page 16, lines 17-19, page 17, lines 10-14; and in Figs. 1-3.

Claim 8 has been canceled

Claims 16-25 have been added. Support for the new claims can be found in the specification at page 8, line 22 to page 19, line 4, and in Figs. 1-3.

No new matter has been added.

Drawings Objections

Applicants have replaced the originally filed informal drawings with the attached formal drawings, and as such, submit that the examiner's objections in the above Office Action are now moot. It is submitted that the drawings are in a condition for allowance.

No new matter has been added.

Specification Objections

The Examiner has objected to the specification due to informalities. Applicants have amended the specification to correct for clerical and/or typographical errors, and now submit that the specification is in a condition for allowance.

No new matter has been added.

Claim Objections

Claim 5 has been amended to correct for antecedent basis errors such that the objection thereto is now moot. No new matter has been added.

35 USC 112 Rejections

The Examiner has rejected claim 12 under 35 USC 112, second paragraph, as containing the trademark UNIX®. Applicants have canceled the limitation

"UNIX®" as such submit that amended claim 12 overcomes the Examiners rejections under 35 USC 112, second paragraph.

No new matter has been added.

35 USC 102 Rejections

The Examiner has rejected claims 1-3, 14 and 15 under 35 USC 102(b) as being anticipated by the IBM Technical Disclosure Bulletin (IBM TDB), "Visual Make Utility", Volume 39, No. 02, February 1996 (Art of Record).

It is submitted that the above IBM Technical Disclosure Bulletin (IBM TDB), "Visual Make Utility" is directed to a prior art program that uses a "classical 'MAKE' utility". (Page 363, ¶ 1, line 1.) It discloses that these conventional *make* utilities are used to determine a set of target and dependent files, and the needs to build the target files. (Page 363, ¶ 1, lines 1-3.) As recognized by the Examiner, this disclosure recites that ""The *make* file can be used in conjunction with a *make* utility to automatically determine which target files are out of date with respect to their dependent files, and therefore rebuild the targets which are out of date". (Page 363, ¶ 1, lines 3-5.)

The IBM TDB discloses that conventional *make* utilities rely on text *makefiles*, which specify the dependencies and build commands. These *makefiles* specify both targets that are real target files and pseudo-targets that are not actually built. (Page 363, ¶ 1, lines 6-10.) In both cases, the IBM TDB discloses that dependent files are listed for both. (Page 363, ¶ 1, lines 10-11.) The *makefile* also

specifies how to build each target; and this is listed either as commands directly below the target dependency listing, or as a generic 'suffix rule'. (Page 363, ¶ 1, lines 11-13.) IBM TDB discloses that conventional *make* utilities have a tool that can parse source files to determine their dependent files, for the purpose of generating the target/dependency relationships. (Page 363, ¶ 1, lines 15-17.)

Applicants submit that this IBM Technical Disclosure Bulletin "Visual Make Utility" is precisely the prior art to which applicant's invention is aimed at improving upon and overcoming the problems associated therewith. (*See,* specification, page 1, line 8 to page 4, line 25.) As recited in the present application, the present invention provides improvements to conventional make utilities that use descripter file containing dependency rules, macros, and suffix rules to instruct *make* to automatically rebuild the program whenever one of the program component files is modified. (Specification, page 1, lines 8-9 and 18-20.) As disclosed in the present application, a conventional *make* file includes two elements: targets, and dependencies, whereby the *make* utility compares the relationships therebetween, particularly the time stamp of each target and its dependencies. (Specification, page 2, lines 1-6.) "However, the *make* utility requires explicitly detailing the source files or target files needing to be changed or updated which can take significant amounts of time and be cumbersome. Also, the *make* command requires building a substantially complete dependency tree before it starts, which can also take an extensive amount of time and computer resources." (Specification, page 2, lines 13-17.)

The present invention improves upon conventional *make* utilities by providing a software utility program for automating selective updates of source code and corresponding object code. (Specification, page 1, lines 5-6.) In so doing, amended claim 1 is directed to a method for updating existing code in a computer program after inputting new code which defines changes to the existing code. The method includes generating a target file list of target files to update, and a dependency file list of files dependent on the target files. The dependency file list of files is then read into a control file, and once therein, selected lines of these files are split into target strings and prerequisite strings. Those target strings having programming language substitutions are appended to a requisition list, while the prerequisite strings are stored in corresponding requisite arrays thereto. An algorithm is executed, whereby this algorithm first matches target files from the target file list with the substituted target string in the requisition list in the control file. The algorithm then updates those matched target files from the target file list if it is determined that the corresponding prerequisite strings, which are stored in the corresponding requisite arrays in the control file, have been updated more recently than the substituted target string.

Applicants submit that the present invention is not anticipated by the IBM TDB. Anticipation is but the ultimate or epitome of obviousness. To constitute anticipation, all material elements of a claim must be found in one prior art source. In re Marshall, 577 F.2d 301, 198 USPQ 344 (CCPA 1978). The IBM TDB does not recite a control file into which the dependency files are read, and then once

therein, selected lines of these files are split into target strings and prerequisite strings. As such, it does not disclose that those target strings having programming language substitutions are appended to a requisition list, while the prerequisite strings are stored in corresponding requisite arrays thereto. The IBM TDB does not disclose executing an algorithm that matches target files from the target file list with the substituted target string in the requisition list in the control file, and then updating matched target files from the target file list if it is determined that the corresponding prerequisite strings, which are stored in the corresponding requisite arrays in the control file, have been updated more recently than the substituted target string, as is presently claimed in amended claims 1-3, 14 and 15. Amended independent claims 14 and 15, which are respectively directed to a computer program product and a program storage device for executing the method of amended claim 1, also include the above limitations that distinguish amended claim 1 from the above cited IBM TDB.

For the above reasons, applicants submit that the claims of the instant invention include limitations not disclosed nor contemplated by the IBM TDB such that the IBM TDB does not anticipate nor render obvious the instant invention.

Section 103 Issues

The Examiner has also rejected claims 4-10 and 12 under 35 USC 103(a) as being unpatentable over the IBM Technical Disclosure Bulletin (IBM TDB), "Visual Make Utility", Vol. 39, No. 02, February 1996 (Art of Record), in view of Auer et al.

(Auer), U.S. Patent No. 6,067,637, while claims 11 and 13 have been rejected under 35 USC 103(a) as being unpatentable over the IBM TDB in view of Auer et al. (Auer), U.S. Patent No. 6,067,637, in further view of Safonov, up 5,892,951.

With respect to claim 4, it is submitted that claim 4 is dependent upon amended claim 1, and as such, includes the limitations of claim 1 that are distinguishable of the IBM TDB. Amended independent claim 5, from which claims 6-7 and 9-13 now depend, also includes the limitations set forth in claim 1 that distinguish claim 1 from the IBM TDB reference.

In particular, amended independent claim 5 is directed to a method for generating changes and updating existing files and code in a computer program. The method includes reading existing target files and dependency files, and then reading a plurality of dependency files that are associated with the target files into a single control file. Within this single control file, selected lines of the dependency files are split into target strings and prerequisite strings. A utility program is then executed that updates the target files and the dependency files associated with these target files. This utility program includes an interpreted scripting language specifying particular characters to search for in the target files and associated dependency files. A requisition list of target strings having interpreted scripting language substitutions is then generated, in addition to the corresponding requisite arrays for the prerequisite strings using the utility program. The target files are then updated by employing a search technique, defined in the utility program, that includes specified target patterns which identify the existing target files being

updated, whereby these existing target are updated if it is determined from such specified target patterns that said prerequisite strings in said control file have been updated more recently than said substituted target string that the prerequisite strings in the control file have been updated more recently than the substituted target string.

Again, as discussed above, the cited IBM TDB reference does not recite a single control file into which the dependency files are read, and then once therein, selected lines of these files are split into target strings and prerequisite strings. The IBM TDB also does not disclose a utility program that updates the target files and the dependency files associated with these target files by using an interpreted scripting language. It also does not disclose generating a requisition list of target strings having interpreted scripting language substitutions, nor corresponding requisite arrays for the prerequisite strings using the utility program. As such, IBM TDB does not disclose or suggest updating existing target files based on specified target patterns, whereby such existing target are updated if it is determined from such specified target patterns that said prerequisite strings in said control file have been updated more recently than said substituted target string that the prerequisite strings in the control file have been updated more recently than the substituted target string, as is currently recited in the pending claims.

The Auer et al. patent does not overcome the deficiencies of the IBM TDB. It is merely directed to reducing the amount of data to be analyzed by expert systems and, in particular, to a system and method for pre-filtering data and storing

the data for more efficient implementation in the expert system. (Col. 5, lines 5-10.) It only discloses a system for and method of pre-filtering data to create and/or maintain data in a working memory such that the data can be more efficiently used by an expert system. (Abstract.) This can be used for managing faults that occur in an expert system, such as faults that occur in connection with the interaction between a central system and a number of remote systems controlled by the central system. (Abstract.)

Auer et al. does not disclose, contemplate or suggest a single control file into which the dependency files are read, and then once therein, selected lines of these files are split into target strings and prerequisite strings. The Auer et al. also does not disclose a utility program that updates the target files and the dependency files associated with these target files by using an interpreted scripting language. It also does not disclose generating a requisition list of target strings having interpreted scripting language substitutions, nor corresponding requisite arrays for the prerequisite strings using the utility program. As such, Auer et al. does not disclose or suggest updating existing target files based on specified target patterns, whereby such existing target are updated if it is determined from such specified target patterns that said prerequisite strings in said control file have been updated more recently than said substituted target string that the prerequisite strings in the control file have been updated more recently than the substituted target string, as is currently recited in the pending claims.
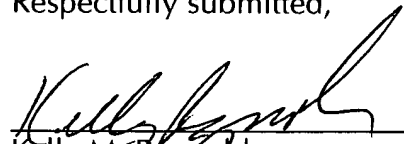
With respect to claims 11 and 13, it is submitted that the cited Safonov patent does not overcome the deficiencies of IBM TDB or Auer et al., alone or in combination. Safonov is limited to a compiler and, specifically, to an LALR syntactical analyzer and semantic checker in a compiler for a high level computer programming language. (Col. 1, lines. 26-29.) It does not disclose, contemplate or suggest a single control file into which the dependency files are read, and then once therein, selected lines of these files are split into target strings and prerequisite strings. Safonov also does not disclose a utility program that updates the target files and the dependency files associated with these target files by using an interpreted scripting language. It also does not disclose generating a requisition list of target strings having interpreted scripting language substitutions, nor corresponding requisite arrays for the prerequisite strings using the utility program. As such, Safonov does not disclose or suggest updating existing target files based on specified target patterns, whereby such existing target are updated if it is determined from such specified target patterns that said prerequisite strings in said control file have been updated more recently than said substituted target string that the prerequisite strings in the control file have been updated more recently than the substituted target string, as is currently recited in the pending claims.

Applicants submit that neither IBM TDB, Auer et al. nor Safonov, taken singly or in any proper combination thereof discloses the instant invention, such that, pending claims 1-7 and 9-25 are neither anticipated by nor rendered obvious over IBM TDB, Auer et al. or Safonov, alone or in any proper combination. It is

submitted that claims 1-7 and 9-25 are properly allowable for the reasons set forth above.

It is respectfully submitted that the application has now been brought into a condition where allowance of the entire case is proper. Reconsideration and issuance of a notice of allowance are respectfully solicited.

Respectfully submitted,

Kelly M. Reynolds
Reg. No. 47,898

DeLIO & PETERSON, LLC
121 Whitney Avenue
New Haven, CT 06510-1241
(203) 787-0595